

IN THE CLAIMS

Please replace all prior versions and listings of claims in the application with the following list of claims:

1. (Currently Amended) A computer-readable medium having computer-executable instructions for performing steps to convert a service discovery stream into an N-ary tree, the service discovery stream having a list of nodes, each node having a data element, a data type and a data size, the service discovery stream having a stream size, the steps comprising:

- a) ~~retrieving~~ determining the data type and the data size of one of the nodes from the service discovery stream, wherein determining the data type of the node comprises identifying the data type from a collection of data types comprising nil, unsigned integer, signed integer, universally unique identifier, text string, Boolean, uniform resource locator, data element sequence and data element alternative data types;
- b) adding the node to a list head;
- c) performing one of decrementing the stream size by the data size and incrementing the service discovery stream to the beginning of a next data element; ~~and~~
- d) obtaining a next node from the list of nodes; and
- e) repeating a), b) and c) until the N-ary tree is formed.

2. (Previously Presented) The computer-readable medium of claim 1 having further computer-executable instructions for performing the steps of:

- ensuring the service discovery stream is well-formed;
- determining the number of nodes in the list of nodes;
- creating a stack; and
- setting the list head to a sibling list pointer of the node.

3. (Original) The computer-readable medium of claim 1 having further computer-executable instructions for performing the step of repeating steps a), b) and c) for the next node.

4. (Original) The computer-readable medium of claim 3 having further computer-executable instructions for performing the step of repeating steps a), b), c) and d) for each node in the list of nodes.

5. (Original) The computer-readable medium of claim 1 having further computer-executable instructions for performing the step of determining if the node is a leaf node.

6. (Original) The computer-readable medium of claim 5 having further computer-executable instructions for performing the steps of:

if the node is a leaf node:

adjusting the service discovery stream beyond the data element;

determining if the stream size of the next node is zero; and

wherein the step of performing one of decrementing the stream size by the element size and incrementing the service discovery stream to the beginning of the next node comprises the step of decrementing the stream size by the data size.

7. (Original) The computer-readable medium of claim 5 having further computer-executable instructions for performing the steps of:

if the node is not a leaf node:

determining if the data size is zero;

if the data size is not zero, the step of performing one of decrementing the stream size by the data size and incrementing the service discovery stream to the beginning of a next node comprises the step of incrementing the service discovery stream to the beginning of the next node; and repeating steps a), b), c), and d).

8. (Original) The computer-readable medium of claim 7 having further computer-executable instructions for performing the steps of:

if the node is not a leaf node:

pushing the list head, the node, and the stream size into a stack; and

setting the list head to one of a sibling list pointer of the node and a container list head;

setting the stream size to one of a size of a parent node content size and a container stream size.

9. (Original) The computer-readable medium of claim 1 having further computer-executable instructions for performing the steps of:

if the stream size is zero;

determining if the stack is empty;

if the stack is not empty:

obtaining a popped list head, the next node, and a popped stream size from the stack;

setting a children pointer of the next node to the list head;

setting the stream size to a revised stream size, the revised stream size formed by

subtracting the data size of the next node from the popped stream size; and

setting the list head to the popped list head.

10. (Currently Amended) A method to convert a service discovery stream to an N-ary tree, the service discovery stream having a first number of elements, each element having an element type and an element size, the service discovery stream having a stream size, the steps comprising:

setting a list head to a sibling list pointer of a first node;

retrieving determining the element type and the element size of a first element from the service discovery stream, wherein determining the element type of the first element comprises identifying the element type from a collection of element types comprising nil, unsigned integer, signed integer, universally unique identifier, text string, Boolean, uniform resource locator, data element sequence and data element alternative element types;

initializing the first node with the first element and the element type and the element size of the first element;

determining if the first node is a leaf node; and

if the first node is a leaf node:

adding the first node to a tail of a list;

adjusting the service discovery stream beyond the first element; and

decrementing the stream size by the element size of the first element; and

obtaining a new node from a node listing.

11. (Original) The method of claim 10 further comprising the steps of
creating an array of nodes having a number of nodes, the number of nodes equal to the
first number of elements plus one; and
creating a stack.

12. (Original) The method of claim 11 further comprising the steps of:

if the first node is a not a leaf node:

determining if the element size of the first node is equal to zero;

if the element size of the first node is not equal to zero:

incrementing the service discovery stream to the beginning of a
next element;

pushing the first node, the stream size, and the list head into a stack;

adding the first node to the tail of the list;

setting the list head to a container list head of the first node; setting the
stream size to a container stream size; and

obtaining the new node from the node listing.

13. (Original) The method of claim 12 further comprising the steps of:

determining if the stream size equals zero; and

if the stream size equals zero:

determining if the stack is empty;

if the stack is not empty:

popping a new list head, the new node, and a new stream size from the stack;

setting a children pointer of the new node to the list head;
setting the stream size to the new stream size minus a size of the new node; and
setting the list head to the new list head.

14. (Original) The method of claim 13 further comprising the steps of: for each node:
- a) retrieving a new element type and a new element size of a new element from the service discovery stream;
 - b) initializing the node with the new element and the new element type and the new element size of the new element;
 - c) determining if the node is a leaf node; and
 - d) if the node is a leaf node:
 - adding the node to a tail of the list head;
 - adjusting the service discovery stream beyond the new element;
 - decrementing the stream size by the new element size of the new element;
 - obtaining the new node from the node listing;
 - e) if the node is a not a leaf node:
 - determining if the new element size of the node is equal to zero;
 - if the new element size of the node is not equal to zero:
 - incrementing the service discovery stream to the beginning of the next element;
 - pushing the node, the stream size, and the list head into the stack;
 - adding the node to the tail of the list;
 - setting the list head to the container list head of the node; setting the stream size to the container stream size;
 - obtaining the new node from the list.
 - f) determining if the stream size equals zero; and
 - g) if the stream size equals zero:
 - determining if the stack is empty;
 - if the stack is not empty:

popping the new list head, the new node, and the new stream size from the stack;
setting a children pointer of the new node to the list head;
setting the stream size to the new stream size minus a size of the new node;
setting the list head to the new list head; and
if the stream size does not equal zero, repeating steps a) to g) for the new node.

15. (Currently Amended) A method to convert a service discovery stream to an N-ary tree, the service discovery stream having a first number of elements, each element having an element type and an element size, the service discovery stream having a stream size, the steps comprising:

setting a list head to a sibling list pointer of a first node of the N-ary tree;
~~retrieving~~ determining the element type and the element size of a first element from the service discovery stream, wherein determining the element type of the first element comprises identifying the element type from a collection of element types comprising nil, unsigned integer, signed integer, universally unique identifier, text string, Boolean, uniform resource locator, data element sequence and data element alternative element types;
initializing the first node of the N-ary tree with the first element and the element type and the element size of the first element;
determining if the first node is a leaf node; and
if the first node is a leaf node:
adding the first node to the list head;
adjusting the service discovery stream beyond the first element;
decrementing the stream size by the element size of the first element; ~~and~~
obtaining a new node from a node listing; and
repeating the acts of setting, retrieving, initializing, and determining until the N-ary tree is formed.

16. (Original) The method of claim 15 further comprising the steps of:
creating an array of nodes having a number of nodes, the number of nodes equal to the
first number of elements plus one; and
creating a stack.

17. (Original) The method of claim 16 further comprising the steps of:
if the first node is a not a leaf node:
determining if the element size of the first node is equal to zero; if the
element size of the first node is not equal to zero:
incrementing the service discovery stream to the beginning of a next element;
pushing the first node, the stream size, and the list head into the stack; adding
the first node to the list head;
setting the stream size to a content size of a parent node;
obtaining the new node from the node listing; and
setting the list head to a sibling list pointer of the new node.

18. (Original) The method of claim 17 further comprising the steps of:
determining if the stream size equals zero; and
if the stream size equals zero:
determining if the stack is empty;
if the stack is not empty:
popping a new list head, the new node, and a new stream size from the
stack;
setting a children pointer of the new node to the list head;
setting the stream size to the new stream size minus a size of the new
node; and
setting the list head to the new list head.

19. (Original) The method of claim 18 further comprising the steps of:

for each node:

- a) retrieving a new element type and a new element size of a new element from the service discovery stream;
- b) initializing the node with the new element and the new element type and the new element size of the new element;
- c) determining if the node is a leaf node; and
- d) if the node is a leaf node:
 - adding the node to the list head;
 - adjusting the service discovery stream beyond the new element; decrementing the stream size by the new element size of the new element; obtaining the new node from the node listing;
- e) if the node is a not a leaf node:
 - determining if the new element size of the node is equal to zero;
 - if the new element size of the node is not equal to zero:
 - incrementing the service discovery stream to the beginning of the next element;
 - pushing the node, the stream size, and the list head into the stack; adding the node to the list head;
 - obtaining the new node from the node listing;
 - setting the list head to the sibling list pointer of the new node;
 - setting the stream size to the container stream size;
 - determining if the stream size equals zero; and
 - if the stream size equals zero:
- f) determining if the stack is empty;
 - if the stack is not empty:
 - popping the new list head, the new node, and the new stream size from the stack;
 - setting a children pointer of the new node to the list head;
 - setting the stream size to the new stream size minus a size of the new node;
 - setting the list head to the new list head; and

if the stream size does not equal zero, repeating steps a) to g) for the new node.

20. (Currently amended) A method to convert a service discovery stream to an N-ary tree comprising:

- a) receiving a reference to a list head, a node list, a stream, and a stream size;
- b) determining if the stream size is equal to zero;
- c) if the stream size is not equal to zero:

~~retrieving~~ determining a data type and a data size of a data element from the stream, wherein determining the data type of the data element comprises identifying the data type from a collection of data types comprising nil, unsigned integer, signed integer, universally unique identifier, text string, Boolean, uniform resource locator, data element sequence and data element alternative element types;

initializing a node with the data type, the data size, and the data element;

adding the node to the list head;

performing one of adjusting the stream beyond the data element and incrementing the stream to the beginning of a next data element;

performing one of decrementing the stream size by the data size and decrementing the stream size by a container size of a stream of a container; ~~and~~

- d) obtaining a next node from the node list; and

- e) repeating a), b) and c) until the N-ary tree is formed.

21. (Original) The method of claim 20 further comprising the steps of:

determining if the node is a leaf node;

if the node is a leaf node, repeating steps a) to c) for the next node; and

wherein the step of performing one of adjusting the stream beyond the data element and incrementing the stream to the beginning of the next data element comprises the step of adjusting the stream beyond the data element and the step of performing one of decrementing the stream size by the

data size and decrementing the stream size by the container size of the stream of the container comprises the step of decrementing the stream size by the data size.

22. (Original) The method of claim 20 further comprising the steps of
determining if the node is a leaf node;
if the node is not a leaf node:

calling a parsing function to receive a reference to the list head of the container, the node list, the stream, and the container size;

repeating steps a) to c) for the next node; and

wherein the step of performing one of adjusting the stream beyond the data element and incrementing the stream to the beginning of the next data element comprises the step of incrementing the stream to the beginning of the next data element and the step of performing one of decrementing the stream size by the data size and decrementing the stream size by the container size of the stream of the container comprises the step of decrementing the stream size by the container size of the stream of the container.

23. (Original) The method of claim 20 further comprising the steps of:

creating an array of nodes having a number of nodes, the number of nodes equal to a number of data elements in the stream;

setting the list head to a sibling list pointer of a first node;

obtaining the next node; and

calling a parsing function to receive a reference to the list head of the container, the node list, the stream, and the container size.